**Sistema de Gestão de Salões de Jogos Arcade**

**Base de Dados**

Turma P11 - GRUPO 7:

* Carlos Verenzuela – 114597
* João Figueiredo - 112930

**Uma imagem com texto, Tipo de letra, Gráficos, design gráfico

Os conteúdos gerados por IA poderão estar incorretos.**

**1. Introdução**

Apresenta o objetivo geral do projeto.

Este projeto tem como objetivo desenvolver um sistema de informação para a gestão de um salão de jogos arcade, contemplando a gestão de funcionários, jogadores, máquinas, sessões de jogo, manutenções, e outras entidades relacionadas. A aplicação integra uma base de dados relacional com uma interface gráfica desenvolvida em C# para facilitar a interação com o sistema.

**Requisitos Funcionais**

1. **Máquinas**:
   * O sistema deve permitir adicionar máquinas de jogos, com informações como tipo (Arcade, Pinball, Simulador), modelo e status.
   * Deve permitir atualizar o status da máquina para "em manutenção" ou "em funcionamento".
   * O sistema deve possibilitar o registo de data de instalação para ajudar no gerenciamento de manutenção e substituição de máquinas.
2. **Jogadores**:
   * O sistema deve permitir o cadastro de jogadores com nome, email, telefone e outras informações pessoais.
   * Deve permitir associar um programa de fidelidade a cada jogador, registando as recompensas, pontos acumulados e atividades realizadas.
   * O histórico de jogos deve ser mantido para personalizar as interações e ofertas para cada jogador.
3. **Funcionários**:
   * O sistema deve registar dados sobre os funcionários do salão, incluindo nome, cargo, turno e salário.
   * Deve ter funcionalidades para definir as responsabilidades dos funcionários, como manutenção de máquinas e/ou organização de evento.
4. **Cartões de Fichas**:
   * O sistema deve permitir a atribuição de cartões de fichas aos jogadores, com informações sobre o número de fichas compradas, usadas e atuais.
   * O sistema deve permitir o acompanhamento do saldo de fichas e a recarga desses cartões.
5. **Sessões de Jogo**:
   * O sistema deve registar cada sessão de jogo, incluindo a máquina utilizada, o tempo de jogo, a pontuação final e o status da sessão.
   * Deve permitir o acompanhamento do desempenho dos jogadores nas diferentes máquinas e gerar relatórios sobre sessões de jogo.
6. **Manutenção das Máquinas**:
   * O sistema deve permitir o registo de manutenção preventiva e corretiva das máquinas.
   * Deve permitir o controle de datas de início e fim da manutenção e o status da manutenção (pendente, concluída).
   * Deve associar a manutenção ao funcionário responsável.
7. **Eventos/Torneios**:
   * O sistema deve permitir a criação e organização de eventos como torneios ou competições.
   * Deve permitir a definição da capacidade de participantes, datas de início e fim, e o status do evento (agendado, concluído).
   * O sistema deve permitir a associação de um funcionário responsável pela organização do evento.
8. **Programas de Fidelidade**:
   * O sistema deve permitir a criação e gerenciamento de programas de fidelidade para jogadores recorrentes.
   * O sistema deve permitir a definição de critérios para acumulação de pontos ou obtenção de descontos, com base no número de jogos ou fichas compradas.
9. **Promoções**:
   * O sistema deve permitir a criação de promoções para os jogadores, como descontos em fichas ou sorteios de prêmios.
   * O sistema deve permitir a associação de promoções a eventos específicos (por exemplo, um torneio de jogos).
   * O sistema deve permitir a definição da duração das promoções e dos critérios de elegibilidade para os jogadores.

**3. Modelação da Base de Dados**

**3.1 Diagrama Entidade-Relacionamento (ER)**

***(Inserir imagem do diagrama ER, que pode ser gerado com ferramentas como dbdiagram.io, Draw.io, MySQL Workbench, etc. Sugere-se mostrar as entidades, atributos e todas as relações com chaves primárias e estrangeiras visíveis.)***

**3.2 Tabelas Principais (Resumo Descritivo)**

**• Maquina(ID\_maquina, Tipo, Modelo, Fabricante, Status, Data\_instalacao)**

**Contém os dados das máquinas de jogo presentes no salão arcade, incluindo o tipo e estado atual. Cada máquina pode estar associada a sessões de jogo e manutenções.**

**• Jogador(ID\_jogador, Nome, Email, Telefone, Pontuacao\_total, ID\_programa\_fidelidade)**

**Representa os clientes que utilizam as máquinas. Estão ligados a sessões de jogo, cartões de fichas e podem usufruir de promoções.**

**• Funcionario(ID\_funcionario, Nome, Cargo, Turno, Salario, Contacto, ID\_manutencao)**

**Funcionários do salão. Alguns podem estar associados a manutenções, carregamentos de cartões ou até à organização de eventos.**

**• Sessao\_jogo(ID\_sessao, Tempo\_jogado, Pontuacao\_final, Status\_sessao, Fichas, ID\_maquina, ID\_jogador)**

**Representa uma sessão de jogo realizada por um jogador numa máquina. Regista a duração, pontuação e número de fichas utilizadas.**

**• Card\_fichas(ID\_card\_fichas, Custo, Fichas\_atuais, Fichas\_usadas, Data\_ultimo\_carregamento, ID\_jogador, ID\_funcionario)**

**Cartões utilizados pelos jogadores para jogar nas máquinas. Cada cartão pode ser recarregado por um funcionário e está associado a um jogador.**

**• Carregamento(Valor, Creditos, ID\_card\_fichas, ID\_funcionario)**

**Representa cada carregamento de um cartão de fichas, indicando o valor pago e os créditos recebidos. É efetuado por um funcionário.**

**• Manutencao(ID\_manutencao, Tipo, Data\_inicio, Data\_fim, Descricao, Status, ID\_maquina, ID\_funcionario)**

**Regista intervenções técnicas nas máquinas, realizadas por funcionários. É usada também para saber se alguma máquina está inativa.**

**• Evento(ID\_evento, Nome, Data\_inicio, Data\_fim, Capacidade, Status, ID\_funcionario)**

**Representa eventos especiais no salão arcade (torneios, festas, etc.) e quem os organiza (funcionário).**

**• Promocao(ID\_promocao, Nome, Tipo, Data\_inicio, Data\_fim, ID\_evento)**

**Promoções associadas a eventos, com prazos definidos. Jogadores podem beneficiar destas promoções.**

**• Usufrui\_promocao(ID\_jogador, ID\_promocao)**

**Tabela associativa que liga jogadores às promoções de que usufruíram.**

**• Programa\_fidelidade(ID\_programa\_fidelidade, Nome, Descricao, Tipo, Requisitos, Data\_inicio, Data\_fim)**

**Programas de fidelidade aos quais os jogadores podem aderir, concedendo vantagens a médio/longo prazo.**

**• Gerente(ID\_gerente, Data\_inicio, Data\_fim, Responsabilidade, ID\_evento)**

**Gerentes são responsáveis por supervisionar eventos e gerir equipas.**

**• G\_Tem\_F(ID\_gerente, ID\_funcionario)**

**Relação entre gerentes e os funcionários sob a sua responsabilidade (tabela N:N).**

**3.3 Relacionamentos e Chaves Estrangeiras**

* **Jogador → Programa\_fidelidade**
* **Card\_fichas → Jogador, Funcionario**
* **Sessao\_jogo → Jogador, Maquina**
* **Manutencao → Maquina, Funcionario**
* **Evento → Funcionario**
* **Promocao → Evento**
* **Usufrui\_promocao → Jogador, Promocao**
* **Carregamento → Card\_fichas, Funcionario**
* **Funcionario → Manutencao (opcional)**
* **G\_Tem\_F → Gerente, Funcionario**
* **Gerente → Evento**

**Todas as relações foram implementadas com restrições de integridade referencial via *foreign keys*.**

**5. Interface Gráfica – Aplicação C#**

**5.1 Tecnologias usadas:**

* .NET Windows Forms
* C#
* SQL Server

**5.2 Funcionalidades:**

* Listagem de funcionários com seleção.
* Visualização detalhada de cada funcionário.
* Edição e atualização com validação.
* Eliminação com confirmação.
* Inserção com campos opcionais.

**5.3 Estrutura de Classes:**

* Funcionario: representa a entidade.
* Funcionarios.cs: formulário principal com listagem, botões e handlers de evento.

**7. Desafios e Soluções**

* **Campos opcionais em sp\_add\_funcionario**: uso de DBNull.Value no C#.
* **Atualização parcial sem sobrescrever com nulls**.
* **Organização da interface para suportar modos visualização/edição**.

**9. Anexos**

**A.1 Scripts SQL:**

* Criação de tabelas.
* Stored procedures (sp\_add\_funcionario, sp\_eliminar\_funcionario, etc.).
* Índices e constraints relevantes.

**A.2 Código C# relevante:**

* Classes Funcionario, métodos buttonAdicionar\_Click, CarregarFuncionarios(), etc.